

LA-UR-21-23497

Approved for public release; distribution is unlimited.

Title: Comparing Emulation Methods for Computer Models with High Dimensional Output

Author(s): Hutchings, Grant Christian
Sanso, Bruno
Gattiker, James R.
Francom, Devin Craig
Pasqualini, Donatella

Intended for: Report

Issued: 2021-04-12

Disclaimer:

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Comparing Emulation Methods for Computer Models with High Dimensional Output

Grant Hutchings, Bruno Sansó, Devin Francom,
James Gattiker, and Donatella Pasqualini

April 5, 2021

Abstract

This Master's Capstone project will present a comparison of statistical models for computer simulation studies. The four models included in this comparison study were chosen for both their proven and diverse methodologies. We will present a case study on hurricane flood data in the Delaware bay which highlights the strengths and weaknesses of each model when applied to a very large spatial field. As computers have gotten faster, we have become interested in modeling increasingly large spatial fields in both size and resolution. Statistical algorithms that are able to efficiently handle these fields have never been more important. We therefore find this comparison to be extremely topical.

Contents

1	Introduction	1
2	Emulation Methods	2
3	Data	3
4	Model Formulation	6
4.1	Simulation Enabled Prediction and Inference (SEPIA)	7
4.2	Bayesian Adaptive Spline Surfaces (BASS)	8
4.3	Bayesian Additive Regression Trees (BART)	8
4.4	Robust Gaussian Stochastic Process Emulation (RobustGaSP)	10
5	Comparison Study	11
5.1	Predictive Accuracy	12
5.1.1	RMSE	12
5.1.2	Inundation	13
5.2	Computational Feasibility	17
6	Sensitivity Analysis	20
6.1	BASS	21
6.2	SEPIA	21
6.3	BART	21
6.4	RobustGaSP	23
7	Discussion	26
8	Appendix	29

1 Introduction

A major challenge in the study of complex physical systems is acquiring sufficient experimental data. Experiments are often expensive, and in many fields, data are limited by the occurrence of natural phenomena. Suppose we wish to understand how a complicated system depends on a number of parameters (inputs). If we define the range of all possible parameter values to be the parameter space, then experimental data is limited by the cost and feasibility of gathering data at any point in the parameter space. If the space is large, an exhaustive search becomes quickly infeasible.

Scientists have turned to simulation as a way to overcome these challenges. Leveraging theoretical knowledge along with available experimental data has led to accurate and reasonably fast computer simulations. Unlike experiment, simulators are not burdened by physical limitations and can be used to explore the system much more efficiently. Computer simulations do not however solve the experimental data problem completely. Depending on the system of interest, simulations can take hours or even days to run at each point in the parameter space making a reasonably exhaustive search of the space once again infeasible. Additionally, simulators are often deterministic, which is a major limitation as uncertainty quantification is very desirable. Statistical models of these computer simulations, referred to interchangeably here as emulators or surrogate models, are designed to solve these problems. They should be very fast and often default to Bayesian methods which provide straightforward uncertainty quantification.

The analysis presented here will compare four emulation methods on hurricane flooding data in the Delaware Bay. These data exhibit some of the challenges mentioned above in that hurricane flood data is not easily gathered at high spatial resolution, and is completely limited by the number and type of hurricanes that make landfall. Simulated data allows researchers to learn about hurricane flood risk on an accelerated timeline and explore hurricanes with any set of parameters. The emulation methods we have chosen implement very different statistical models, all of which have been proven themselves a reasonable choice for similarly structured spatial data. The goals

of this comparison study are to quantify the accuracy of predictions, understand the computational requirements, and compare the sensitivity analysis options given by each method. Sensitivity analysis is important in this problem because researchers are very interested in understanding which hurricane parameters are most influential for determining inland flooding. Some kinds of sensitivity analysis allow priors to be specified for those parameters, and prior specific sensitivities to be compared. In Section 2 we will give a brief overview of the methods included in our study and explain why they were chosen.

2 Emulation Methods

Simulation Enabled Prediction and Inference (SEPIA) (Gattiker et al., 2020b) implements the Gaussian Process model described in Higdon et al. (2008). This model was originally implemented at Los Alamos National Laboratory as the MATLAB code GPMSA (Gattiker et al., 2020a) and in 2020 was refurbished and translated to python. SEPIA makes use of a basis representation of the data to fit a Gaussian Processes (GP) to each of the basis coefficients. This is a tried and true methodology for spatial modeling that has seen much success in the literature.

Our implementation of Bayesian Adaptive Spline Surfaces (BASS) (Francom and Sansó, 2020) similarly makes use of a basis representation, but takes a wholly different approach to modeling basis coefficients. Rather than using GP's, adaptive splines are used. BASS has been recently applied to large spatial data from computer experiments and has shown great results (see, for example, Francom et al., 2019).

The implementation considered in this work of Bayesian Additive Regression Trees (BART), developed in Sparapani et al. (2021), once again makes use of a basis representation. Basis coefficients are fit using an additive regression tree model. Treed models have seen success in the literature for their speed and flexibility, and BART has proven to be effective in similar settings such as a recent analysis of airborne particulate data over California (Zhang

et al., 2020). BART was also compared to BASS in Francom et al. (2019) and was shown to be very comparable.

The fourth method considered in this work consists of Robust Gaussian Stochastic Process Emulation (RobustGaSP) (Gu et al., 2017) which provides a way to circumvent the basis decomposition by fitting a GP to each point in space. This is made computationally feasible by both parallel computation, and the assumption of shared range parameters for all GP’s. RobustGaSP does not make use of MCMC for model fitting like the other three models. Instead parameters are fit using numerical optimization of marginal posterior distributions. These major model differences make this an interesting inclusion to our comparison study. RobustGaSP has also shown promising results on large scale computer model emulation of large volcanic flow simulations (Gu and Berger, 2016).

These models were chosen because they have all been shown to produce accurate inference using quite different methodologies. We will give a more detailed description of each model in Section 4 and then compare them on simulated hurricane flooding data in the Delaware Bay in Section 5. The goal of this comparison is to highlight the strengths and weaknesses of each model formulation (and implementation) in a big data setting. We are interested in how predictions from these models compare in terms of classical statistical measures of accuracy such as root mean squared error (RMSE), but also application specific measures of flooding which can be used to better understand hurricane induced flood risk in the Delaware Bay.

3 Data

The Sea, Lake and Overland Surges from Hurricanes (SLOSH) simulator is a computer code developed by the National Weather Service to estimate storm surge heights from hurricanes. Storm surge height is defined as the maximum water height due to the hurricane at any single location. Our data consists of an ensemble of 4,000 runs from the SLOSH simulator. Each storm in the ensemble is defined by a unique set of input parameters: sea level rise,

heading, velocity, minimum air pressure, and latitude. Figure 1 presents a spatial map of one of these runs.

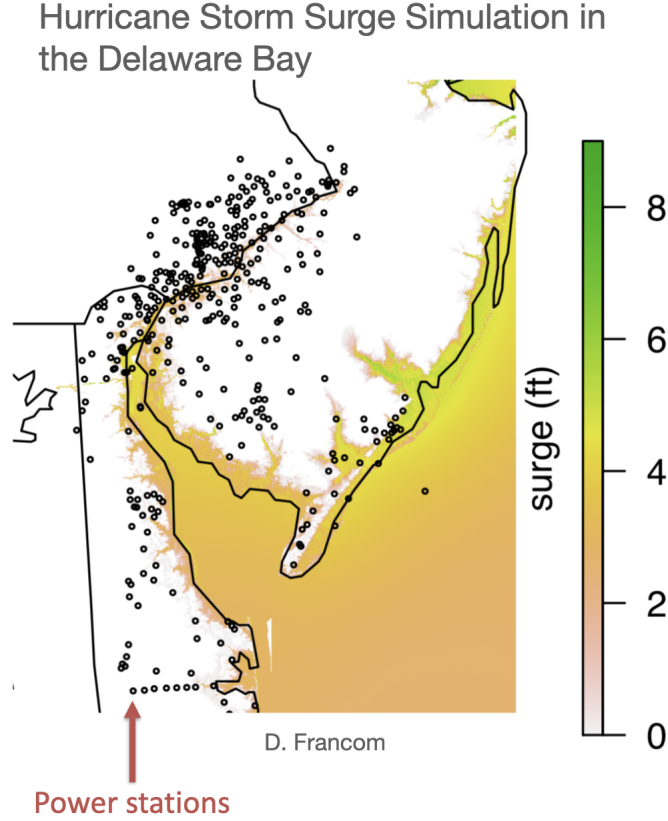


Figure 1: SLOSH output

Input parameters for the ensemble were determined using a uniform Latin hypercube design over the parameter space. Models are trained on a subset of this ensemble and tested on storms outside of the training set. Below we show the parameter space for a training set of 500 storms and its associated 50 storm testing set. We can see that the parameters are uniformly distributed over their domains.

One output from SLOSH is a $4,520 \times 5,115$ grid with the storm surge height at each of these 23,119,800 locations. If we ignore the curvature of the Earth, we have a spatial resolution of approximately 0.06 mi lat x 0.05

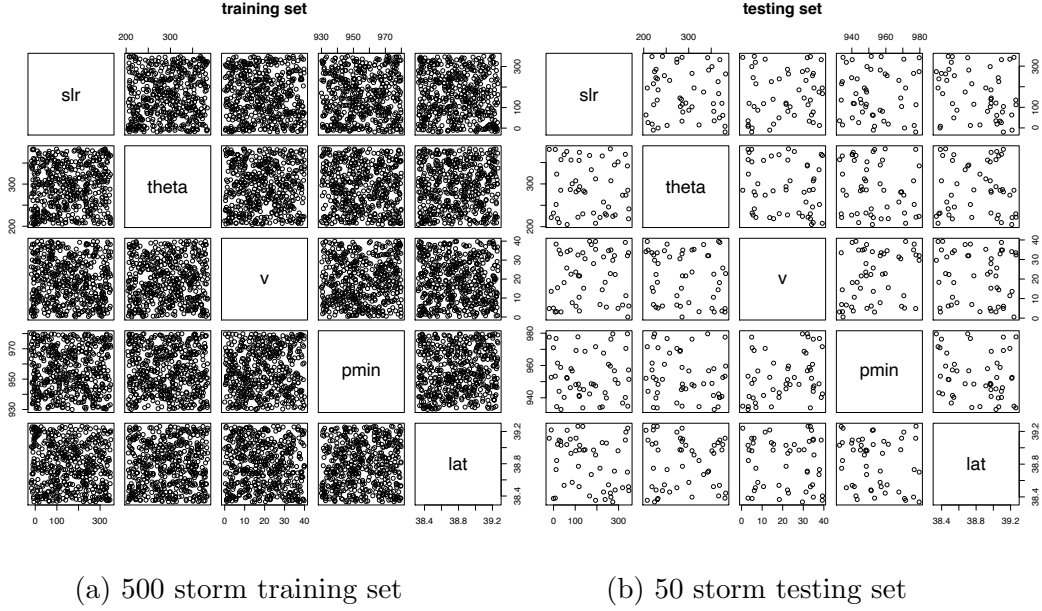


Figure 2: Pairs plot of input parameters

mi long. This many spatial locations present a formidable computational challenge. This is fortunately eased by the fact that the majority of the points are far enough inland that there is no flooding for any of the 4,000 simulations. By modeling only cells which take non-zero values in at least one of the simulations we reduce the size of the field to 3,500,000 locations.

Accurate prediction of flooding is important for a variety of reasons including displacement of residents, and property/infrastructure damage. One area of specific interest for this project is flood prediction near power stations, which are displayed as black dots in Figure 1. A storm surge of four feet or higher is considered catastrophic for a power station. We are interested in the emulators' ability to accurately predict a surge height of greater than four feet, as this information is very valuable for determining if a power station should preemptively shut down due to an approaching storm.

4 Model Formulation

Computer model emulators have many desirable properties, perhaps the most notable being accurate out of sample prediction. But accuracy is not everything, we care about other aspects of the model such as computational complexity. Surrogate models should be faster than the simulator itself. If this is not the case, we may only be gaining uncertainty quantification by choosing emulation over simulation. Our goal here is to compare four very different emulation methods on the grounds of predictive accuracy and computational complexity. We will explain the pro's and con's of each model and present results from our case study to quantify the differences between the models.

The main challenge in modeling this data is the size. Our emulator must be able to handle 4,000 runs from SLOSH, each with 3,500,000 response values. One very common approach to reduce the dimension of a problem like this is to decompose the data into principal components using a singular value decomposition.

Let m be the number of storms in the ensemble, and k the number of spatial locations. Then we can define $\mathbf{s} = \{s_1, \dots, s_k\}$ to be the set of locations, and $\mathbf{X} \in \mathbb{R}^{m \times p}$ the matrix of input parameters where p is the dimension of the inputs.

Then the response $\mathbf{Y}(\mathbf{X}, \mathbf{s}) \in \mathbb{R}^{m \times k}$ is approximated as

$$\mathbf{Y}(\mathbf{X}, \mathbf{s}) \approx \mathbf{W}(\mathbf{X})\mathbf{B}(\mathbf{s}) \quad (1)$$

where $\mathbf{B}(\mathbf{s}) \in \mathbb{R}^{J \times k}$ is a matrix of basis vectors where each of the J rows corresponds to a basis vector for the j^{th} principal component $j = 1, 2, \dots, J$. Each element in that row corresponds to one of the k spatial locations. $\mathbf{W}(\mathbf{x}) \in \mathbb{R}^{m \times J}$ is a matrix of basis vector coefficients where each of the J columns holds the m basis coefficients for the principal components. The expended form of Equation (1) is

$$\begin{bmatrix} y_{11} & y_{12} & \dots & y_{1k} \\ y_{21} & y_{22} & & \vdots \\ \vdots & & \ddots & \\ y_{m1} & \dots & & y_{mk} \end{bmatrix} \approx \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1J} \\ w_{21} & w_{22} & & \vdots \\ \vdots & & \ddots & \\ w_{m1} & \dots & & w_{mJ} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1k} \\ b_{21} & b_{22} & & \vdots \\ \vdots & & \ddots & \\ b_{J1} & \dots & & b_{Jk} \end{bmatrix} \quad (2)$$

For the purposes of modeling, $\mathbf{W}(\mathbf{X})$ determined by the decomposition become the new response values, taking the place of the original response $\mathbf{Y}(\mathbf{X}, \mathbf{s})$ with the spatial information contained in $\mathbf{B}(\mathbf{s})$. An emulator modeling $\mathbf{W}(\mathbf{X})$ is completely ignorant of the original response $\mathbf{Y}(\mathbf{X}, \mathbf{s})$ and the target of inference is to determine J basis coefficient functionals $w_j(\mathbf{x})$ using the m response values for each j .

J determines the accuracy of our approximation to \mathbf{Y} , and is usually chosen so that a fixed percentage of the variation in the data is accounted for by the decomposition. For this SLOSH ensemble, we are able to represent 99% of the variance using only 25 basis vectors. That means we are fitting 25 uni-variate models rather than one model with a 3,500,000 dimensional response, a significant reduction in complexity. This approach makes models like Gaussian Process which scale $O(n^3)$, far more feasible, which will be made clear in Section 5.2 when we look at model fit times.

For emulators making use of this decomposition, we will assume the data to be mean zero as it is centered and scaled prior to determining the principal component decomposition. We will now give a more detailed description of each emulator.

4.1 Simulation Enabled Prediction and Inference (SEPIA)

SEPIA is a python code developed by Jim Gattiker, Natalie Klein, Grant Hutchings and Earl Lawrence at Los Alamos National Lab and implements the model described in Higdon et al. (2008). By utilizing the decomposition in Equation (1) a Gaussian Process is fit to each basis function coefficient $w_j(\mathbf{x})$ where \mathbf{x} is any set of input parameters.

$$w_j(\mathbf{x}) \sim GP(0, \sigma_j^2 C(\mathbf{x}, \mathbf{x}' | \boldsymbol{\phi}_j)) \quad (3)$$

Where σ_j^2 is the marginal variance, and $C(\mathbf{x}, \mathbf{x}' | \boldsymbol{\phi}_j)$ the covariance matrix defined for the vector of correlation distances $\boldsymbol{\phi}_j$. Independent Gamma and Beta priors are given to the marginal variances and correlation distances respectively. Given σ_j^2 and $\boldsymbol{\phi}_j$ a zero mean normal prior is defined for each w_j . For a full model specification including hyperpriors, refer to Higdon et al.

(2008). The resulting posterior distributions are explored via MCMC. This methodology is frequently used at Los Alamos National Lab for surrogate modeling.

4.2 Bayesian Adaptive Spline Surfaces (BASS)

BASS was developed by Devin Francom at UCSC as part of his PhD Thesis with Bruno Sansó. The following model formulation outline was taken from Francom and Sansó (2020). Similar to SEPIA, BASS makes use of a basis representation to handle large amounts of data. BASS models each w_j as

$$w_j(\mathbf{x}) = a_0 + \sum_{m=1}^M a_m Z_m(\mathbf{x}) \quad (4)$$

where a_0, a_1, \dots, a_M are constants and Z_1, \dots, Z_M are basis functions learned from the data. The basis functions have the form

$$Z_m(\mathbf{x}) = \max(0, \prod_{k=1}^{K_m} g_{km}[s_{km}(x_{v_{km}} - t_{km})]^\alpha) \quad (5)$$

where $s_{km} \in \{-1, 1\}$ is the sign, $t_{km} \in [0, 1]$ is a knot, v_{km} selects a covariate, K_m is the degree of interaction and $g_{km} = [(s_{km} + 1)/2 - s_{km}t_{km}]^\alpha$ is a constant that makes the basis function have a maximum of one. The exponent α defines the degree of the polynomial splines. Note that variables can only be used once in each basis function.

To fit this model we need to estimate $\boldsymbol{\theta} = \{\sigma^2, M, \mathbf{a}, \mathbf{K}, \mathbf{s}, \mathbf{t}, \mathbf{v}\}$. This is done via a reversible jump MCMC algorithm. For specifics on priors and the RJMCMC algorithm see Francom and Sansó (2020).

4.3 Bayesian Additive Regression Trees (BART)

Tree models, such as Treed GP (Gramacy and Lee, 2008) have proven to be fast, flexible, and accurate for computer emulation problems. BART is a treed model that has been very reliable in other case studies of this nature. Recently, BART was found to be very effective for the spatial modeling of

Ambient fine particulate matter pollution (PM_{2.5}) over California (Zhang et al., 2020). As detailed in Chipman et al. (2010), BART is a sum of trees model where scalar output $w_j(\mathbf{x})$ is approximated as

$$w_j(\mathbf{x}) = \sum_{i=1}^I g(\mathbf{x}|T_i, M_i) + \epsilon, \quad \epsilon \sim N(0, \sigma^2) \quad (6)$$

where each T_i is a regression tree and \mathbf{x} is a p dimensional vector of inputs $\mathbf{x} = (x_1, \dots, x_p)$. Each tree can incorporate one or more of the p inputs, corresponding to main and interaction effects. Each tree node has a binary decision rule to determine whether parameter $\{\theta \in A\}$ or $\theta \notin A$. $M_i = \{\hat{\theta}_{1i}, \dots, \hat{\theta}_{bi}\}$ represents the set of b terminal node parameter estimates. The function g maps $\{\hat{\theta}_{1i}, \dots, \hat{\theta}_{bi}\} \in M_i$ to \mathbf{x} .

This additive structure endows BART with a high amount of flexibility when the number of trees is large. This does however come at the price of complexity. BART needs to estimate $\{(T_1, M_1), \dots, (T_I, M_I), \sigma\}$ for I trees where T_i and M_i are not single parameters, but an entire tree structure fit with a set of decision rules, and a set of terminal nodes respectively.

BART utilizes a backfitting MCMC algorithm for posterior sampling where all parameters are gibbsable. This is key for keeping computation time down as the number of parameters scales. The flexibility BART can provide is therefore very desirable as the additional computational cost is reasonably low. The key to enforcing this flexibility is a regularization prior which forces the effect from each tree to be small. This prevents individual tree effects from dominating the additive structure.

With posterior draws $(T_1^*, M_1^*), \dots, (T_I^*, M_I^*)$ prediction is straightforward,

$$f^*(\cdot) = \sum_{i=1}^I g(\cdot|T_i^*, M_i^*) \quad (7)$$

(Sparapani et al., 2021).

4.4 Robust Gaussian Stochastic Process Emulation (RobustGaSP)

RobustGaSP was developed by Mengyang Gu as part of his PhD thesis at Duke. Unlike the other three methods, parameters are estimated using marginal likelihood optimization rather than MCMC. This has its drawbacks when it comes to uncertainty quantification as confidence bounds must be estimated using distributional assumptions, but MCMC has many challenges which are avoided using this approach.

RobustGaSP implements a computationally feasible alternative to the Many Single (MS) emulation approach (Conti and O’Hagan, 2010; Lee et al., 2011, 2012). Individual emulators are fit to each coordinate, which in the context of our case study means 3,500,000 individual Gaussian Process emulators. Each emulator has its own mean function and variance, but they all share range parameters γ , which are estimated from the joint likelihood of all emulators (Gu and Berger, 2016).

Using our previous notation, define m runs of the simulator output at k locations to be $\mathbf{Y}(\mathbf{X}, \mathbf{s})$ where \mathbf{X} is the matrix of input parameters. Then

$$y_i(\mathbf{x}) \sim GP(\boldsymbol{\mu}(\mathbf{x}), \sigma^2 c(\mathbf{x}, \mathbf{x}')), ; i = 1, \dots, k \quad (8)$$

where $\boldsymbol{\mu}(\mathbf{x})$ is a mean function, σ^2 is the variance, and $c(\mathbf{x}, \mathbf{x}')$ is the correlation function. Then for each location we have a multivariate likelihood

$$(y_i(\mathbf{x}_1), \dots, y_i(\mathbf{x}_m) | \boldsymbol{\mu}, \sigma^2, \Sigma) \sim \mathbf{MVN}((\mu_{x_1}, \dots, \mu_{x_m}), \sigma^2 \Sigma) \quad (9)$$

The mean function is modelled as a simple linear regression with parameters $\boldsymbol{\theta}$ and prior

$$\pi(\boldsymbol{\theta}, \sigma^2) \propto \frac{1}{\sigma^2} \quad (10)$$

A key to this model is the jointly robust (JR) prior applied to the range parameters $\gamma = \{\gamma_1, \dots, \gamma_p\}$ where p is the dimension of the input parameter vector. This prior was introduced in Gu (2018) and is called jointly robust because it cannot be written as the product of marginal priors and its robust in marginal posterior mode estimation.

First consider reparameterizing to the inverse range parameters $\beta = 1/\gamma$. Then the JR prior is defined as

$$\pi^{JR}(\beta_1, \dots, \beta_p) = C_0 \left(\sum_{l=1}^p C_l \beta_l \right)^\alpha \exp \left\{ -b \left(\sum_{l=1}^p C_l \beta_l \right) \right\}, \quad (11)$$

where $C_0 = \frac{(p-1)! b^{a+p} \prod_{l=1}^p C_l}{\Gamma(a+p)}$, $a > -(p+1)$, $b > 0$ and $C_l > 0$ are parameters. As we will discuss in Section 6, this prior facilitates the form of sensitivity analysis provided by the package.

The posterior distribution resulting from this model formulation is marginally optimized to obtain parameter estimates.

5 Comparison Study

We will compare these emulators in terms of their predictive accuracy as well as computational feasibility when applied to the 4000 run ensemble of hurricane flooding data from SLOSH. Computation time for each emulator will generally be a function of training set size, but will vary between emulators as each algorithm has different scaling properties. We would like to be able to train our models with as few storms as possible. This not only means minimal computation time, but also leaves more holdout storms for validation of the model. We consider seven different training set sizes; 50, 100, 500, 1000, 1750, 2500, and 3636 storms. 3636 was chosen as the largest training set size because it is the largest number that permits a testing set size of 10% (364 testing storms). Training and testing storms were selected as follows; First, randomly select 3,636 storms from the 4000 as our master training set, leaving 364 as our master testing set. For each smaller training and testing set we choose storms randomly from the master sets. For example, to build a training set of 100 storms, we randomly draw 100 storms from the set of 3636 training storms, then draw the associated 10 testing storms from the master testing set of 364 storms. Our comparison study involves training each of the four emulators on each of the training sets, and computing all prediction metrics for each testing set. This will allow us to determine optimal training set size, and compare emulators both within and between training set size.

BASS, BART, and SEPIA all make use of MCMC for parameter estimation. For each model we collect 10,000 MCMC samples and burn the first 9000 leaving 1000 available for prediction. Given the size of the spatial field, we were forced to thin these 1000 samples down to 50 due to memory constraints on our computing resources. Consider our largest testing set of 364 storms and suppose we wish to generate predictions using all 1000 posterior samples. This requires a matrix of size $(364 \times 3,500,000 \times 1000)$. In R, a double precision value requires 8 bytes. A matrix of this size requires approximately 10 terabytes of storage. We are therefore limited to a more modest 500 gigabyte matrix resulting from the use of 50 MCMC samples. This is one of the challenges that comes with a data-set of this size. One possible way to use more samples for prediction, which we do not consider here, is to limit the area of interest for prediction to a subset of the spatial field. For example we could look at only cells surrounding power stations, or simply only look at small swaths of land one at a time.

5.1 Predictive Accuracy

This comparison will make use of raw predictions of storm surge height as well as other metrics of inundation (flooding). For our uses inundation will refer to the area of catastrophic flooding, defined as the number of land locations with more than 4 feet of flood water, as well as the associated flood volume. Flood volume is computed as the sum of water heights over all catastrophically flooded locations. We will present results for the root mean squared errors (RMSE) as well as summaries of area and volume predictions compared to values from the holdout set.

5.1.1 RMSE

There are two relevant questions associated with RMSE; first, are there any consistent trends among the simulators independent of training set size? And second, is it necessary to train a model on all 3,636 storms to achieve good performance? We will use Figure 3 to attempt to answer these questions. The figure shows boxplots of RMSE for each emulation method at each training

set size at which they were run. Each individual RMSE value is for one storm in the out of sample testing set and is computed using the mean over samples from the posterior predictive distributions. Notice that the maximum training set size for SEPIA and RobustGaSP are 1000 and 500 respectively. This is due to computational limitations which will be described in Subsection 5.2.

We would like to highlight a few things from Figure 3. As expected, RMSE is generally decreasing with training set size. It is important to note that this decrease suffers from diminishing returns. It would appear that a training set size of greater than 1000 runs is unnecessary to achieve near optimal RMSE.

The other thing we notice is that the differences between emulators at each training set size is not extreme. Differences are more extreme in smaller training sets, but with increasing set size, emulators seem to be producing very similar RMSE's. Assuming we would choose at least 500 training storms, all of the emulators give similar results. It is harder to make conclusions regarding the differences at set sizes of 50 and 100 as differences between the emulators may be in part due to the relatively small testing sets.

5.1.2 Inundation

Inundation, or flooding, will be measured using both the area and volume of catastrophic flooding. Area is simply the total number of land cells with a water level greater than 4 feet, and volume is the sum of the water levels for these cells. We limit our predictions to land locations as these metrics are irrelevant at sea.

We compare models trained on 500 storms as we have data for all the emulators at this level. Looking at Figures 4, and 5 each 'test_storm_id' on the x-axis is one storm from the testing set and the y-axis represents flood area (volume) with red dots to indicate true data values. We see that predictions are generally quite good, and interestingly the emulators tend to over or under-predict the same storms. This could be an indication that certain points in the parameter space are hard to emulate regardless of which

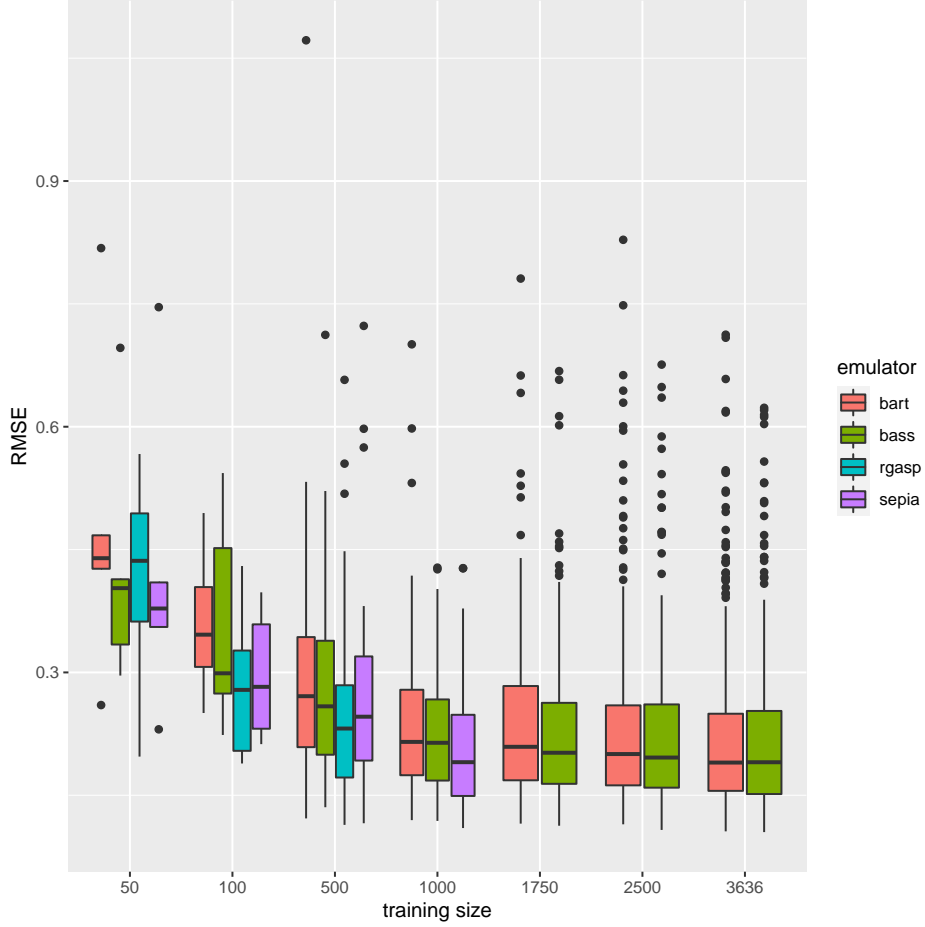


Figure 3: RMSE by training set size

method you choose. We also note that RobustGaSP is more likely to cover the true data value, but this is due to the fact that it estimates very large uncertainty compared to the other models. Figures 13, 14, 15, and 16 in the Appendix show similar plots for training set sizes of 1000 and 3636.

Figures 4, and 4 present information on the level of individual storms. Figures 6, and 7 show the RMSE for area and volume predictions over all testing storms. The RMSE for the area predictions is presented on the log scale so that differences between the emulators can be seen more clearly.

We notice that RobustGaSP seems to do worse when we look at RMSE

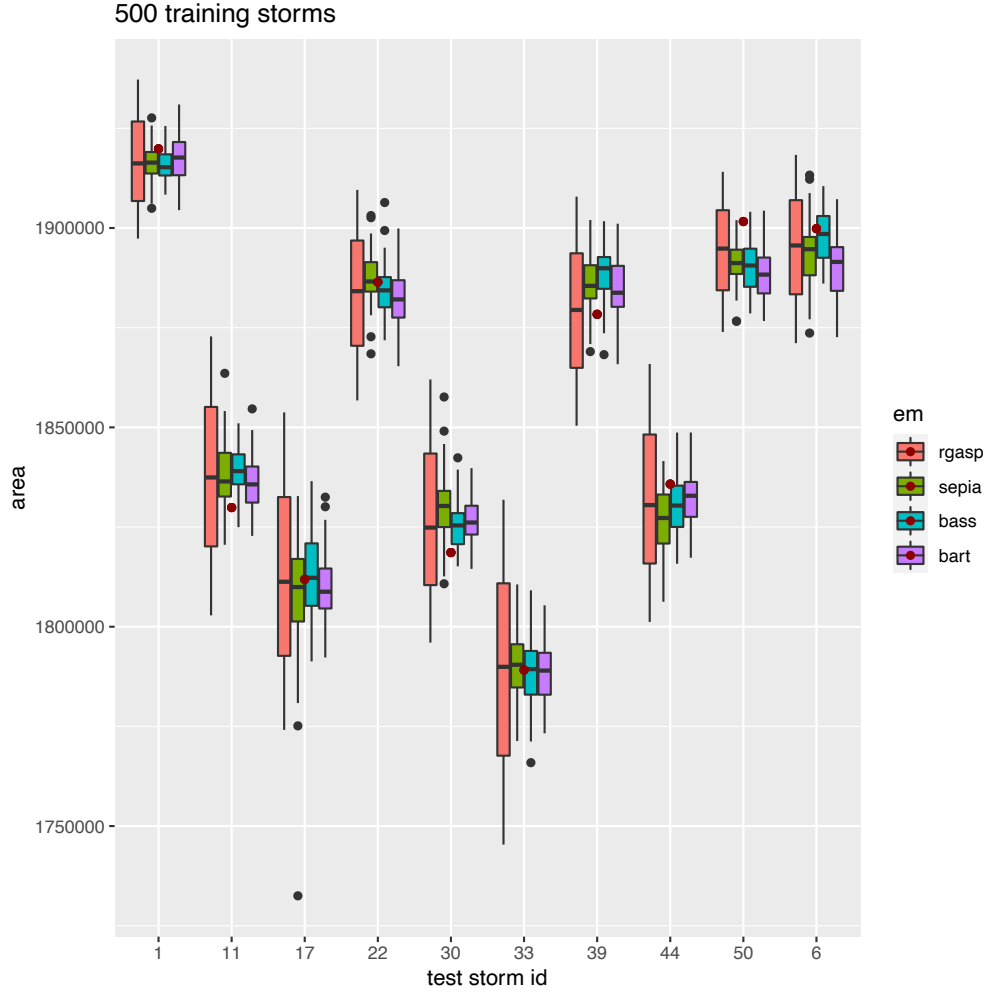


Figure 4: Flood area predictions for models fit to 500 storms. Dark red dots indicate true flood area from SLOSH.

than when we look at raw predictions. For raw predictions, RobustGaSP has mean area and volume predictions are very comparable and sometimes better the other emulators. The poor RMSE results occur because the prediction intervals are so wide. The mean error is pulled up due to the conservative upper and lower bounds on the predictions.

BASS is consistently very good with surprisingly good results for small

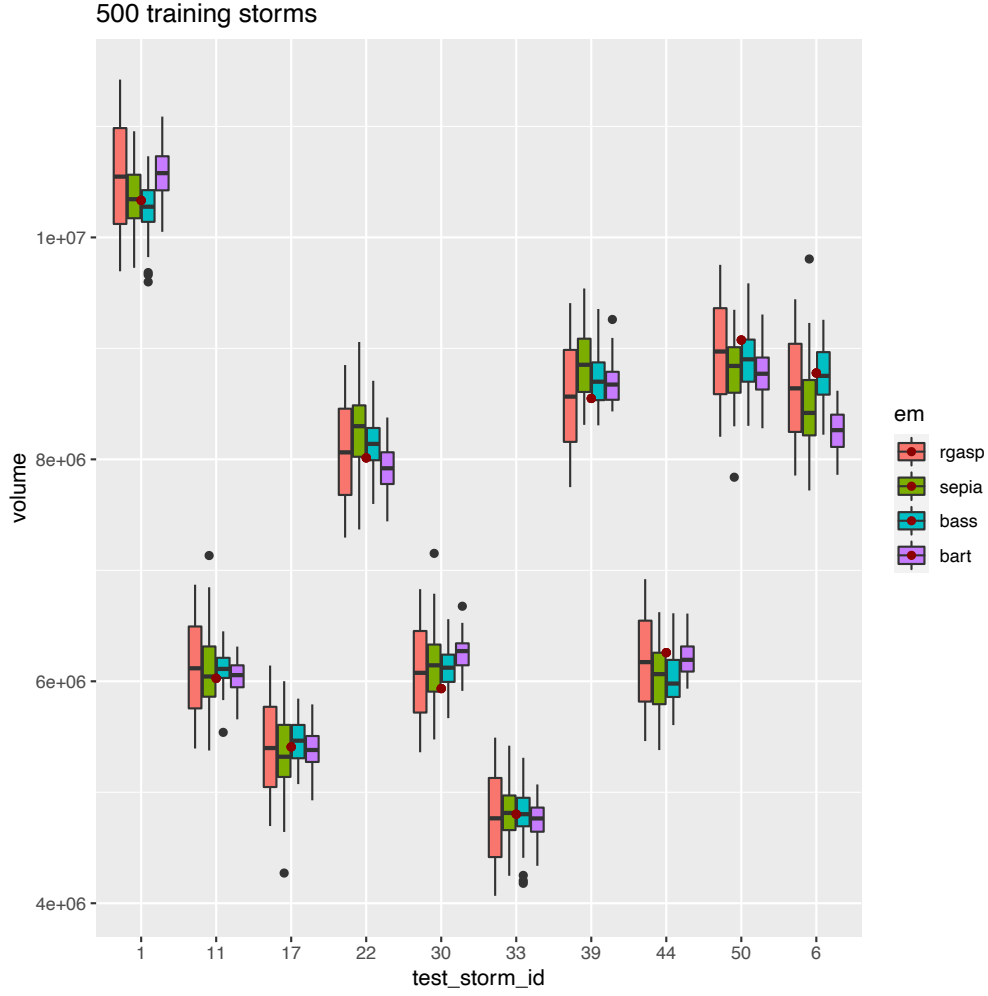


Figure 5: Flood volume predictions for models fit to 500 storms. Dark red dots indicate true flood volume from SLOSH.

training sizes. Similar to RMSE for surge height predictions, we see diminishing returns when it comes to training set size and RMSE reduction. It seems that a training set of 1000 storms is sufficient.

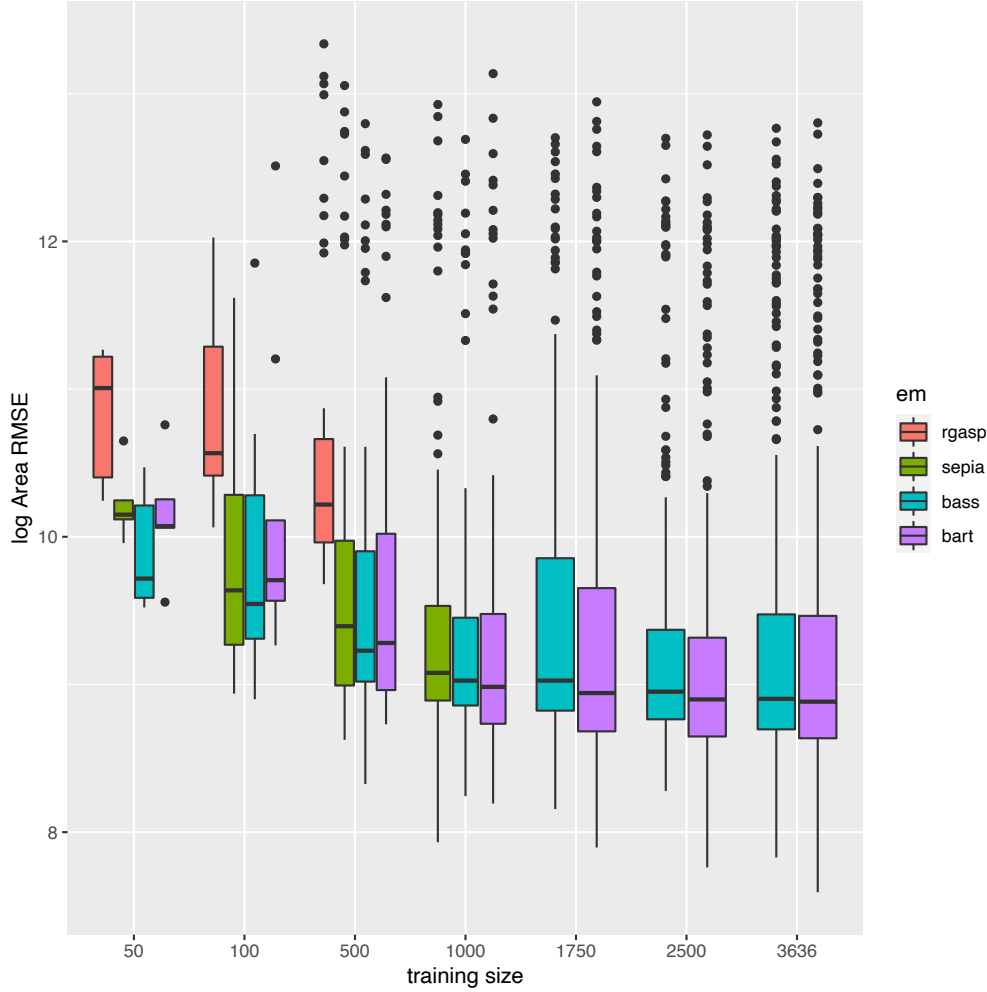


Figure 6: Area prediction RMSE on log scale.

5.2 Computational Feasibility

Computation time is an important aspect of any comparison of emulation methods especially on large data sets where some methods are simply not feasible. SLOSH takes about eight minutes per storm to run, which is not especially long for a simulator. Computation time for an emulator is therefore very important in this case study. All of our models were built on a Los Alamos compute cluster 1.5TB node with 96 cores, 2 Xeon Platinum 8260

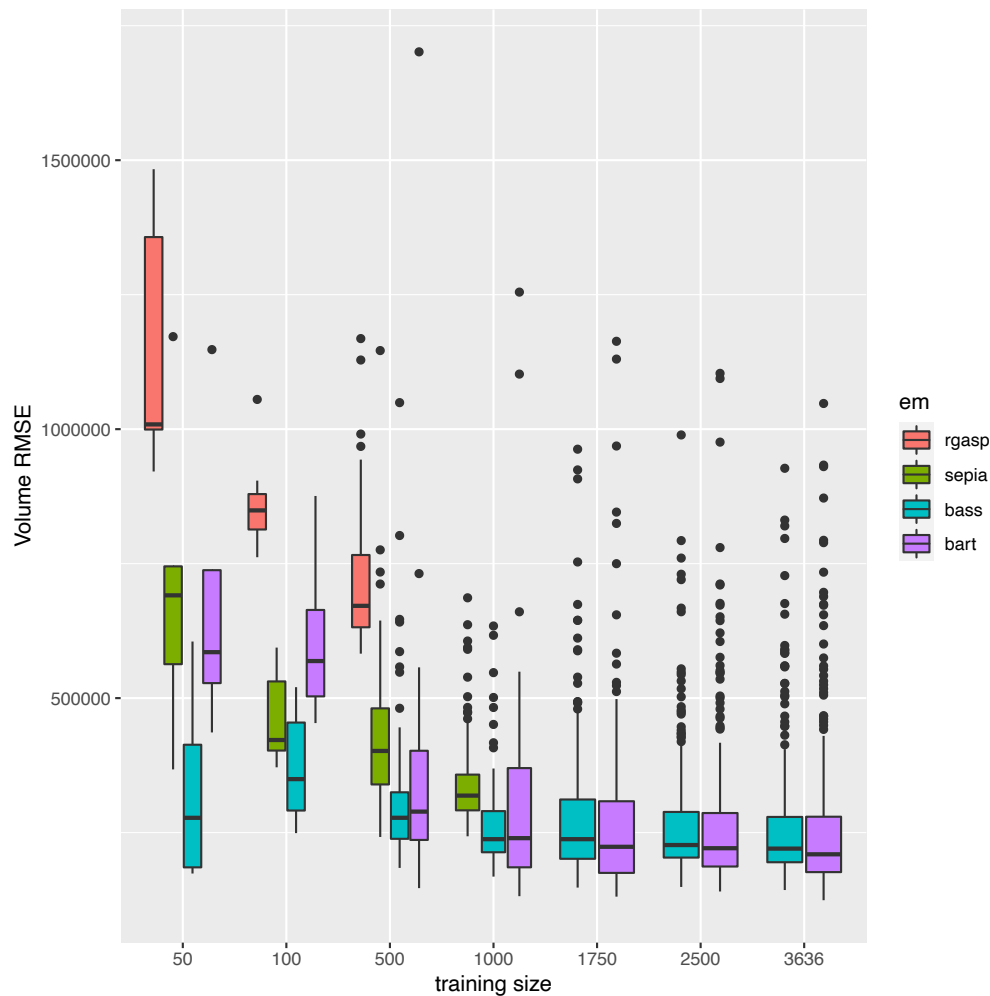


Figure 7: Volume prediction RMSE.

CPUs @ 2.40GHz, and 192GB of Dynamic RAM. Jobs on this node are limited to 48 hours.

What we found is that BASS remains quite fast, even with the full training set size of 3,636 storms. With this training set it takes only 34 seconds to fit the model. BART is also reasonably fast at 318 seconds. BASS and BART remain fast as the training set size increases because they scale approximately linearly.

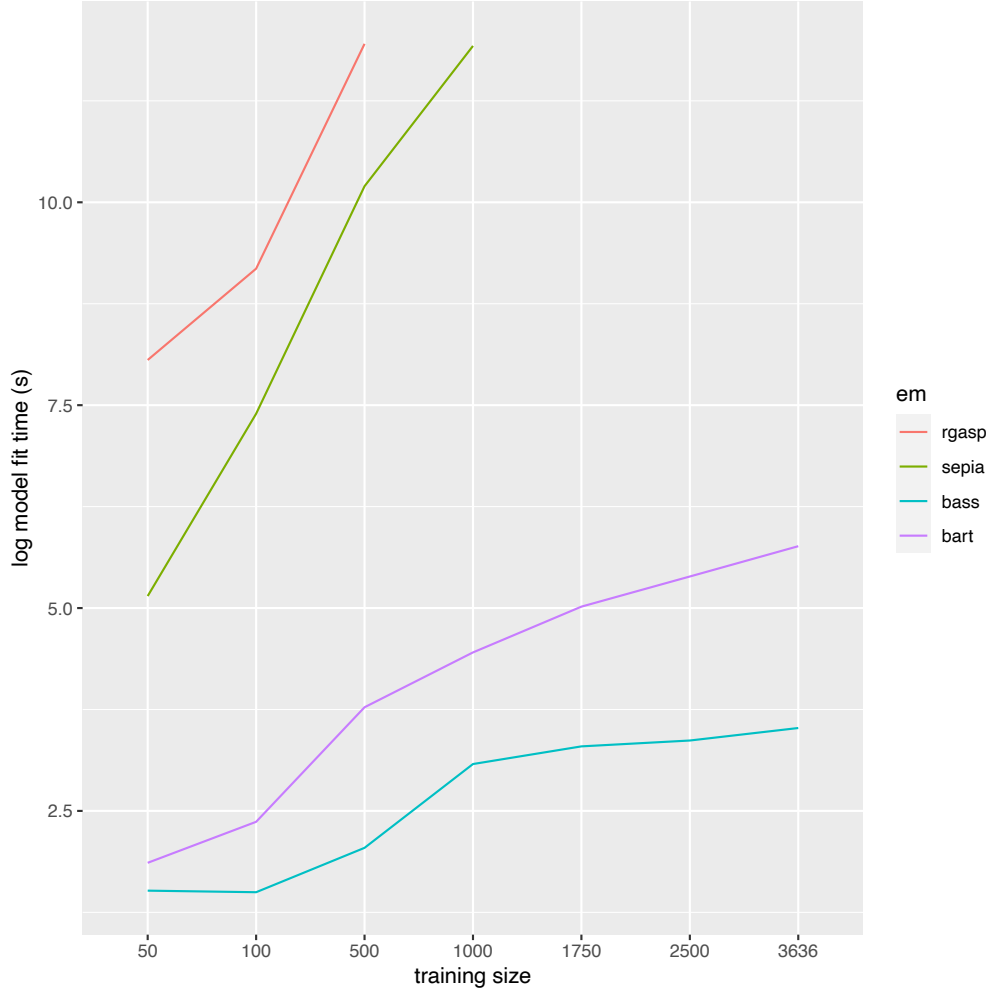


Figure 8: Model fit time

SEPIA and RobustGaSP do not share this linear scaling. Both methods make use of Gaussian Process which is limited by $O(n^3)$ scaling. Given our 48 hour time limit we were only able to fit SEPIA with a maximum training set size of 1000 storms, which took nearly the full 48 hours. It should be mentioned that one of the reasons for this is that SEPIA's parallel MCMC algorithm is currently still under development. However, even if we make use of an embarrassingly parallel MCMC and split our MCMC up into say

10 chains, reducing computation time by an order of magnitude, we will still never be able to compete with BASS and BART.

For RobustGaSP we are limited to 500 storms, again reaching our time limit of 48 hours. We should not be too surprised that RobustGaSP is the slowest of the four methods given the scope of the optimization problem it attempts to tackle; fitting 3,500,000 Gaussian Processes.

Fortunately for RobustGaSP and SEPIA, we believe that 3,636 training storms is not necessary to achieve near optimal prediction. We have seen that RMSE for surge height, flood area, and flood volume all level off around 1000 training storms.

6 Sensitivity Analysis

Sensitivity analysis in computer models consists of determining which inputs have the greatest (least) effect on the response. Quantifying the percentage of the variability in the response due to each input is done through functional analysis of variance (ANOVA) (Gu, 2018). Sobol indices, computed using draws from the posterior predictive distribution, provide this information (Sobol, 2001). An additional, very desirable property of Sobol indices is that different priors can be considered for the model inputs and sensitivities can be compared across these prior assumptions. This is very applicable to our case study as hurricane priors are location specific, and are not unanimously agreed upon.

SEPIA and BASS have built in functionality to compute these indices, BART and RobustGaSP do not. Methods for computing Sobol indices have been generalized in the R package 'sensitivity' (Iooss and Pujol, 2021), so in theory sensitivity indices are available for all four models. However, the sensitivity package requires multiple draws from the predictive distribution. RobustGaSP does not have this functionality available to the user. One would need to use the parameter estimates from the model to manually simulate realizations from the Gaussian Processes which is a drawback of the implementation. Rather than do that, we will describe the form of sensitivity

analysis that RobustGaSP has built in. BART has a uniquely different challenge in terms of a Sobol decomposition. Since we are fitting and combining uni-variate BART models on the basis coefficients, the sensitivity package will only provide sensitivity for the basis coefficients, not the original response. We cannot simply aggregate these results into an overall sensitivity. In the following Subsections we will detail the sensitivity analysis provided by each emulator.

6.1 BASS

As we mentioned, BASS provides us with Sobol indices. In Figure 9 we plot main effect Sobol indices colored by the square root of the explained variance. We can see that sea level rise explains most of the variation in the data. We also see that velocity is most important at the northern opening to the bay, and has a reasonable effect all along the northern coast.

We can also get sensitivity indices for interaction effects. Below we plot two example interaction effects. We can see that interactions between sea level rise and minimum pressure play an important role in the furthest inland flooding. As our goal here is not to analyze these sensitivities, rather to demonstrate the information provided by the Sobol decomposition, we used simple uniform priors over the input parameters.

6.2 SEPIA

SEPIA also has built in functionality for computing Sobol indices which provides sensitivities for the original response, not just the basis coefficients. Unfortunately, we found the implementation unable to handle data of this size.

6.3 BART

BART offers a unique form of sensitivity analysis by keeping track of the number of times each input variable shows up in a regression tree. For every posterior predictive sample, we calculate the percentage of trees containing

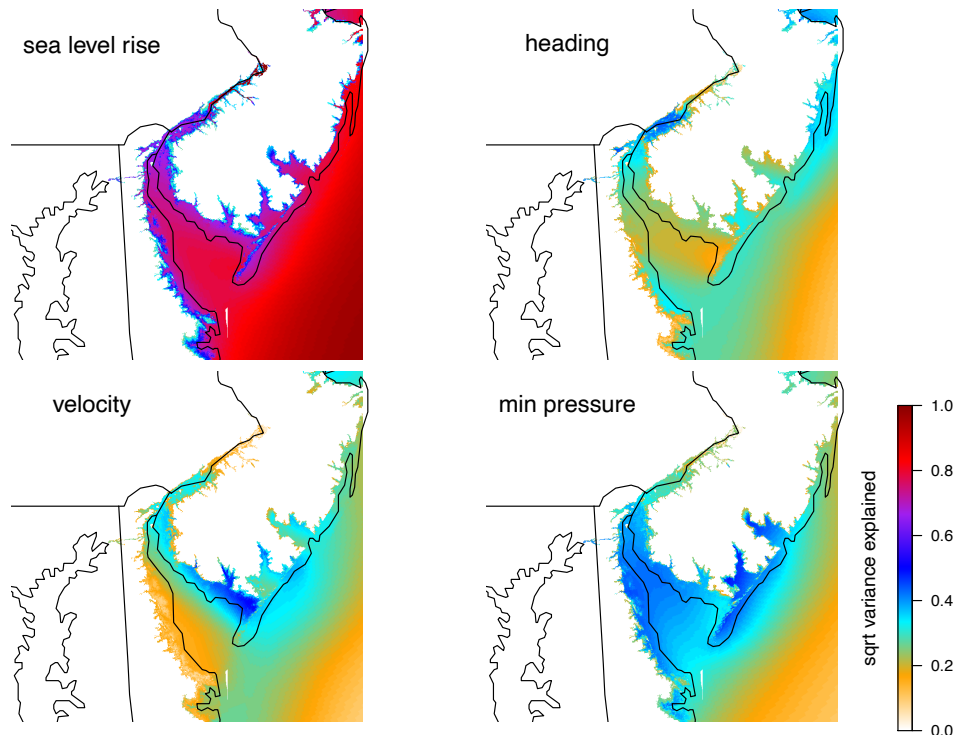
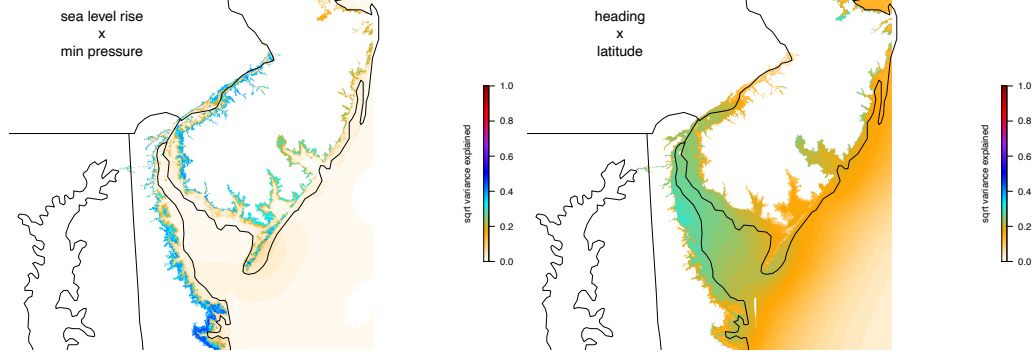


Figure 9: BASS: Main effects sensitivity analysis

each input variable. This gives a distribution of percentages over posterior draws. The drawback is that information is only available for individual models corresponding to a single principal components coefficient. We cannot simply aggregate over components to get sensitivity for the original response. As an example of why this would not be informative, consider that the last principal component accounts for a very small percentage of the variance in the response. We should not consider frequencies of input variables from this component to be nearly as important as those from the first few components. This kind of sensitivity is therefore more useful when only a single BART



(a) Sea Level Rise x Minimum Pressure (b) Heading x Landfall Location

Figure 10: BASS - Interaction effects

model is being considered to represent the response. Figure 11 shows these distributions for the first four principal components. What we see is that Sea level rise (slr) is the dominant input for the first principal component, which accounts for the most variability in the data. These plots may be more useful when combined with visualizations of the principal components. Notice that heading (θ) appears to be very important in both PC2 and PC3. If we look at these principal components in Figure 12 we notice that they explain variability mostly near the coast with PC2 capturing variability inside the bay, and PC3 on the coast between 39 and 40 degrees latitude. Combining information from these figures gives us an idea of the locations in space where heading is an especially important input. We do not show PC1 simply because it is completely dominated by sea level and shows no interesting structure.

6.4 RobustGaSP

RobustGaSP determines if an input is believed to be inert, or contributes little to response variability. Inertness is decided through the estimated range parameters $\hat{\gamma}$. This method of variable selection was introduced in (Linklet-

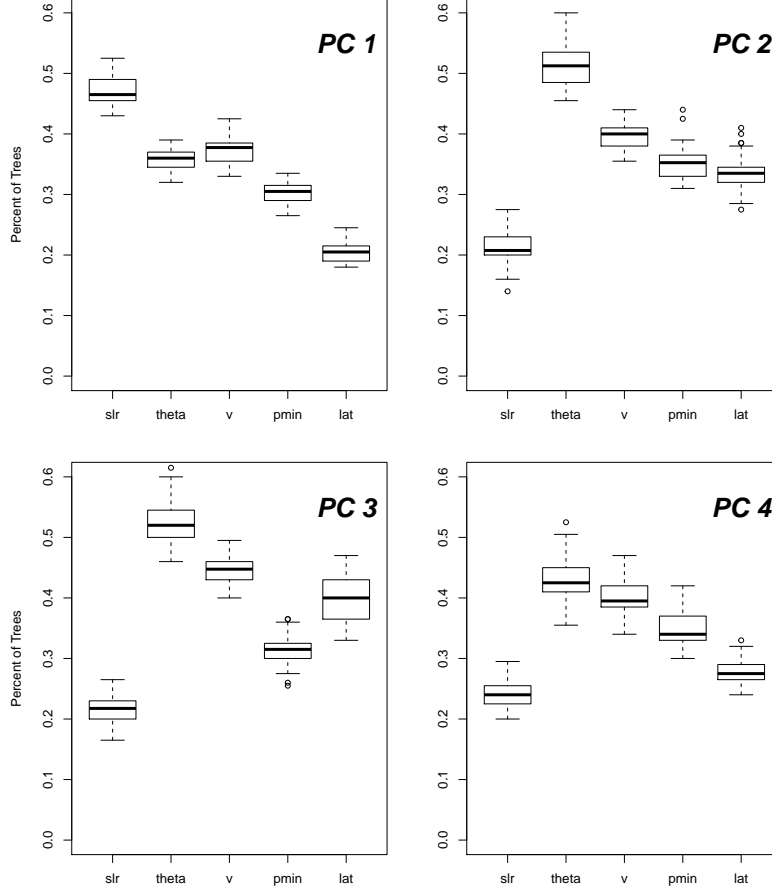


Figure 11: BART: Percentage of trees containing each input variable. Distributions over posterior predictive draws.

ter et al., 2006). If γ_l is inert, $\hat{\gamma}_l \rightarrow \infty$ and has little effect response variability (Gu, 2018). The JR prior we described in Section 4.4 is required for this to work. The key is that this prior, unlike the reference prior, makes sure the marginal posterior for $\gamma > 0$ even if some $\hat{\gamma}_l \rightarrow \infty$. To decide whether a $\hat{\gamma}_l$ is sufficiently large to consider the associated input inert, we consider the normalized inverse

$$\hat{P}_l = \frac{C_l \hat{\beta}_l}{\sum_{i=1}^{p_x} C_i \hat{\beta}_i} \quad (12)$$

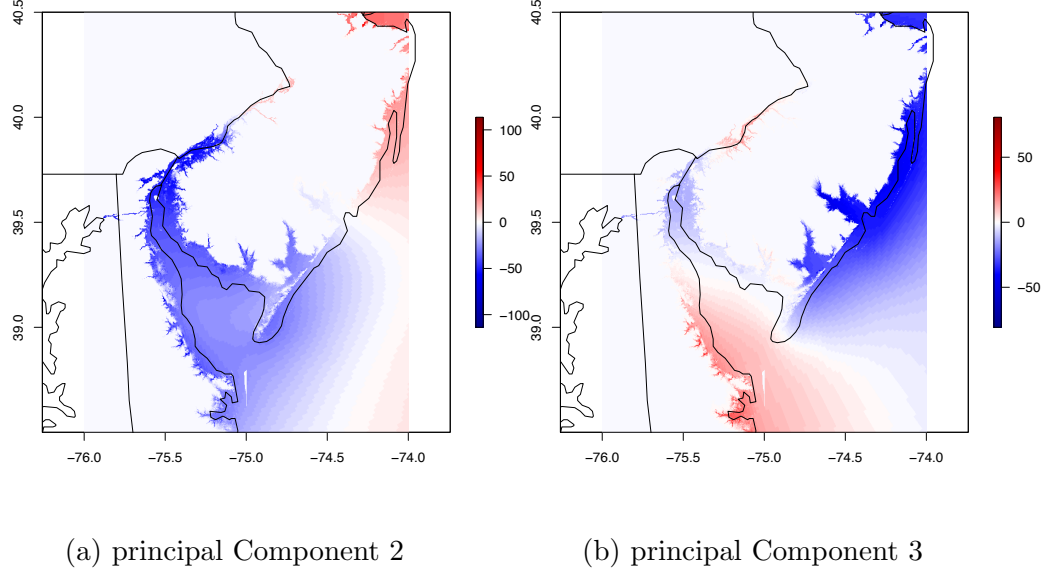


Figure 12: principal Components Spatially

where $\beta_l = 1/\gamma_l$ and C_l is a normalization constant to account for the different scales of the inputs (Gu, 2018). We can then set a threshold (default 0.1) below which an input is determined to be inert. Table 1 shows the results for our storm surge analysis. We see that none of the inputs are found to be inert.

Table 1: Estimated normalized inverse range parameters

sea level rise	heading	velocity	min pressure	latitude
0.7205615	2.2591919	0.9885197	0.5257090	0.5060178

Albeit far less informative from a sensitivity analysis point of view than a Sobol decomposition, this is valuable information which comes for free as a byproduct of the model fit.

7 Discussion

Computer model emulation is most beneficial when applied to simulator that is especially expensive to run. While the SLOSH simulator is not especially expensive, we do not believe this to diminish this comparison of methods presented here. There exist hurricane simulators that are far more complex but provide similarly structured flooding data to SLOSH. It is reasonable to assume that a comparison of these emulators on similar data from a more complex simulator would produce similar results. Additionally, a simulator like SLOSH allows us to generate a large ensemble, which makes our comparison of model fits by training set size possible. We can apply the knowledge gained from this study to more expensive simulators in that an ensemble of more than 1000 runs is likely unnecessary to achieve near optimal predictive performance.

Figures 4, and 5 show that accurate predictions can be obtained all four emulators with training sets of only 500 or 1000 storms. The differences between the models are more apparent when we consider computation time and sensitivity analysis.

Given the 48 hour fit time on a high performance computer for SEPIA with 1000 training storms and RobustGaSP with 500 training storms, we believe these emulators are ill-suited for a problem of this size. SEPIA is most useful when the size of the data is relatively small. In these cases the computational complexity of a Gaussian Process is not prohibitive. RobustGaSP did very well emulating TITAN2D, but the size of the ensemble was only 50 runs. RobustGaSP may be better suited for problems where the ensemble size is small. We note however that even when we trained on a modest 50 storms from SLOSH, RobustGaSP did not outperform the other emulators.

Given the prohibitive computation time of SEPIA and RobustGaSP we consider BART to be the only real competitor with BASS. We do however strongly recommend BASS as it provides a few key benefits over BART. First, it is approximately an order of magnitude faster than BART over all training sets. Second, sensitivity analysis is important for this application. We would like to be able to analyze sensitivities over a wide range of prior assumptions

on the input parameters. We therefore prefer an emulation method that can provide us with Sobol indices (BASS). Lastly, BASS does surprisingly well at out of sample prediction when trained on a small number of storms as seen in Figure 5. This would prove very useful when emulating a more complex simulator where only a small ensemble of runs is available.

In future work we would like to confront some of the questions and limitations that arose during this study. One of which is the outliers in RMSE as seen in Figures 3, 6, and 7. We would like to know if these storms live in a particular region of the parameter space and if they are consistent across methods. One of the limitations that comes with data of this size is the storing of large matrices. This resulted in a small number of posterior predictive samples used for comparison. If storing more samples is not feasible, we would like to convince ourselves that each model has sufficiently converged and our samples are a good representation of the posterior distributions. For SEPIA, BASS, and BART, this means an analysis of the MCMC results, and for RobustGaSP we would like to run the optimization with a number of different starting points to convince ourselves that we are not in a local mode. In Section 5 we discussed the possibility of reducing the area of predictive interest so that we can store more posterior samples. This is something we would like to explore in the future as we have special interest in locations near power stations.

References

- Chipman, H. A., George, E. I., and McCulloch, R. E. (2010). Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1):266–298.
- Conti, S. and O’Hagan, A. (2010). Bayesian emulation of complex multi-output and dynamic computer models. *Journal of Statistical Planning and Inference*, 140(3):640 – 651.
- Francom, D. and Sansó, B. (2020). BASS: An R package for fitting and per-

- forming sensitivity analysis of Bayesian adaptive spline surfaces. *Journal of Statistical Software*, 94(8):1–36.
- Francom, D., Sansó, B., Bulaevskaya, V., Lucas, D., and Simpson, M. (2019). Inferring atmospheric release characteristics in a large computer experiment using bayesian adaptive splines. *Journal of the American Statistical Association*, 114(528):1450–1465.
- Gattiker, J., Higdon, D., and Williams, B. (2020a). lanl/gpmsa.
- Gattiker, J., Klein, N., Hutchings, G., and Lawrence, E. (2020b). lanl/sepia: v1.1.
- Gramacy, R. B. and Lee, H. K. H. (2008). Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483):1119–1130.
- Gu, M. (2018). Jointly robust prior for gaussian stochastic process in emulation, calibration and variable selection.
- Gu, M. and Berger, J. O. (2016). Parallel partial gaussian process emulation for computer models with massive output. *Ann. Appl. Stat.*, 10(3):1317–1347.
- Gu, M., Wang, X., and Berger, J. O. (2017). Robust gaussian stochastic process emulation.
- Higdon, D., Gattiker, J., Williams, B., and Rightley, M. (2008). Computer model calibration using high-dimensional output. *Journal of the American Statistical Association*, 103:570–583.
- Iooss, Sebastien Da Veiga, A. J. and Pujol, G. (2021). *sensitivity: Global Sensitivity Analysis of Model Outputs*. R package version 1.24.0.
- Lee, L., Carslaw, K., Pringle, K., and Mann, G. (2012). Mapping the uncertainty in global ccn using emulation. *ATMOSPHERIC CHEMISTRY AND PHYSICS*, 12:9739–9751.

- Lee, L. A., Carslaw, K. S., Pringle, K. J., Mann, G. W., and Spracklen, D. V. (2011). Emulation of a complex global aerosol model to quantify sensitivity to uncertain parameters. *Atmospheric Chemistry and Physics*, 11(23):12253–12273.
- Linkletter, C., Bingham, D., Hengartner, N., Higdon, D., and Ye, K. Q. (2006). Variable selection for gaussian process models in computer experiments. *Technometrics*, 48:478 – 490.
- Sobol, I. (2001). Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and Computers in Simulation*, 55(1):271–280. The Second IMACS Seminar on Monte Carlo Methods.
- Sparapani, R., Spanbauer, C., and McCulloch, R. (2021). Nonparametric machine learning and efficient computation with Bayesian additive regression trees: The BART R package. *Journal of Statistical Software*, 97(1):1–66.
- Zhang, T., Geng, G., Liu, Y., and Chang, H. H. (2020). Application of bayesian additive regression trees for estimating daily concentrations of pm2.5 components. *Atmosphere*, 11(11).

8 Appendix

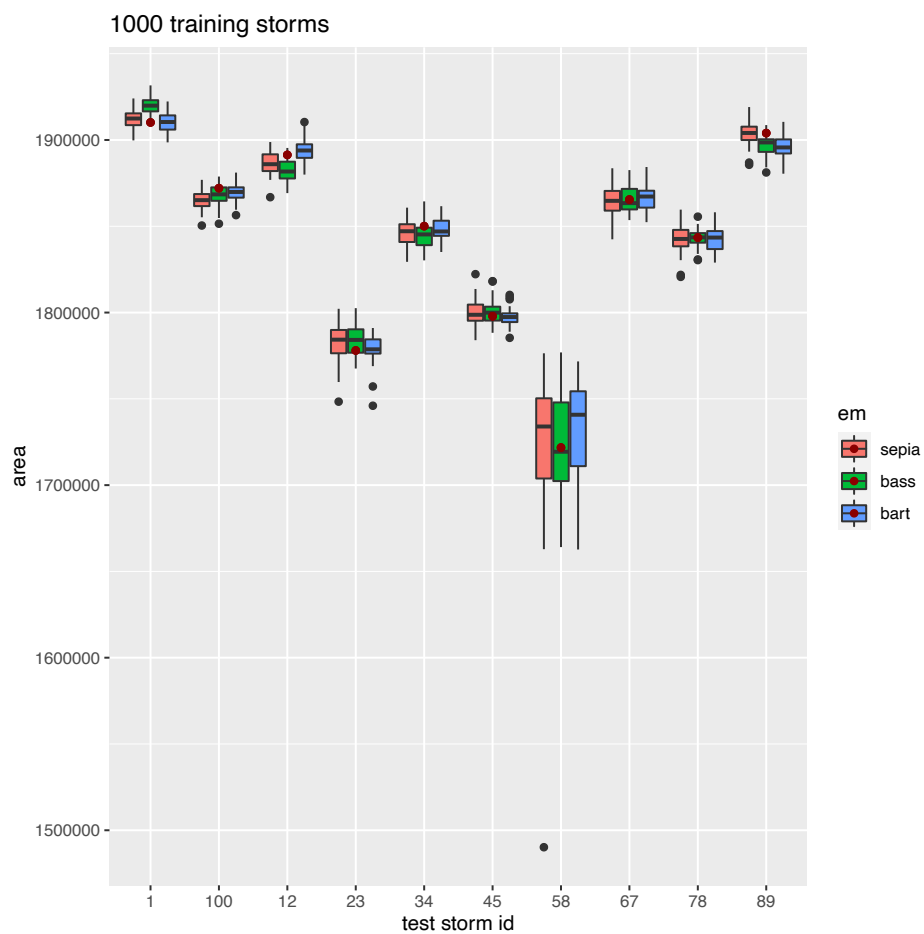


Figure 13: Flood area predictions for models fit to 1000 storms. Dark red dots indicate true flood area from SLOSH.

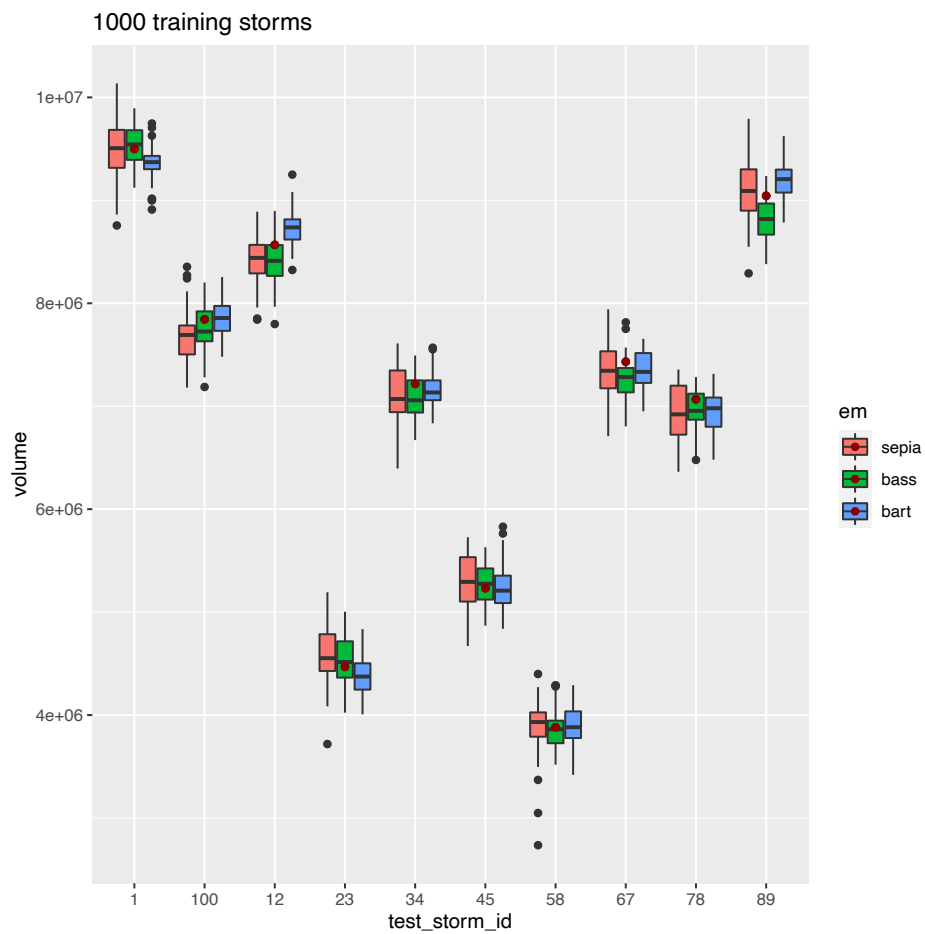


Figure 14: Flood volume predictions for models fit to 1000 storms. Dark red dots indicate true flood volume from SLOSH.

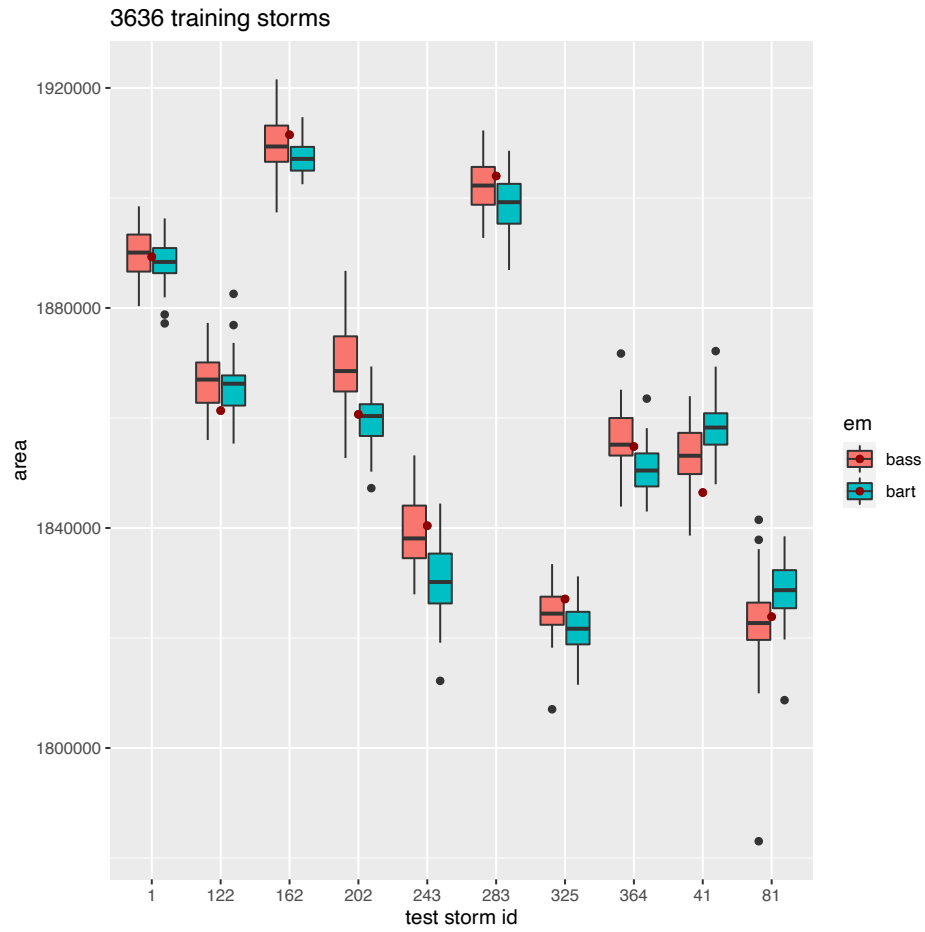


Figure 15: Flood area predictions for models fit to 3636 storms. Dark red dots indicate true flood area from SLOSH.

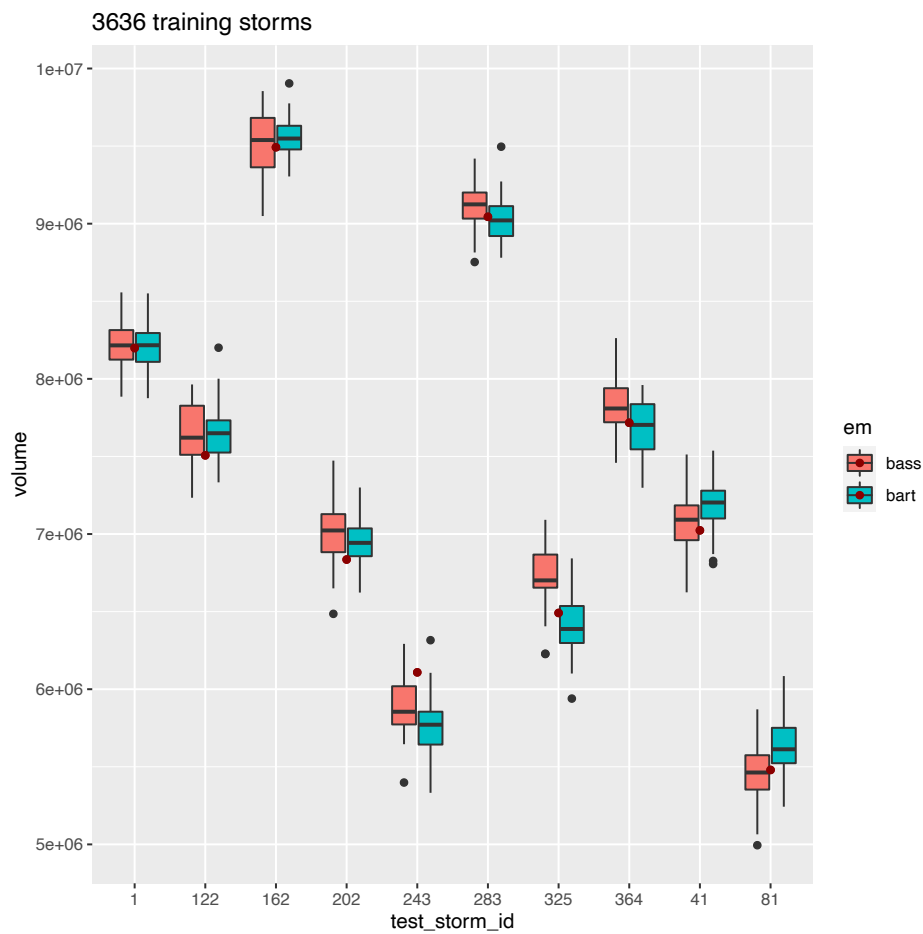


Figure 16: Flood volume predictions for models fit to 3636 storms. Dark red dots indicate true flood volume from SLOSH.